Solution 1: L1 Regularization

(a) (i) $\lambda\|\boldsymbol{\theta}\|_1$ is not differentiable at $\boldsymbol{\theta} = \mathbf{0}$.

(ii) $0.5\|\mathbf{X}\boldsymbol{\theta} - y\|_2^2$ is convex (since it is a quadratic form) and $\lambda\|\boldsymbol{\theta}\|_1$ is also convex (since it is a norm). Since the sum of convex functions is convex it follows that $\mathcal{R}_{\text{reg}}$ is convex.

(iii)

$$\begin{aligned}\frac{\partial}{\partial\theta_j}0.5\|\mathbf{X}\boldsymbol{\theta} - y\|_2^2 &= \frac{\partial}{\partial\theta_j}0.5\sum_{i=1}^{n}\left(y^{(i)} - \sum_{k=1}^{p}\boldsymbol{x}_k^{(i)}\theta_k\right)^2 \\ &= -\sum_{i=1}^{n}\boldsymbol{x}_j^{(i)}\left(y^{(i)} - \sum_{k=1}^{p}\boldsymbol{x}_k^{(i)}\theta_k\right) \\ &= \underbrace{-\sum_{i=1}^{n}\boldsymbol{x}_j^{(i)}\left(y^{(i)} - \sum_{k\neq j}^{p}\boldsymbol{x}_k^{(i)}\theta_k\right)}_{=:\rho_j} + \theta_j\underbrace{\sum_{i=1}^{n}\left(\boldsymbol{x}_j^{(i)}\right)^2}_{=:z_j}.\end{aligned}$$

(iv)

$$\partial_{\theta_j}\mathcal{R}_{\text{reg},\boldsymbol{\theta}_{\neq j}}(\theta_j) = \begin{cases}\{-\rho_j + \theta_j z_j - \lambda\} & \text{for } \theta_j < 0 \\ [-\rho_j - \lambda, -\rho_j + \lambda] & \text{for } \theta_j = 0 \\ \{-\rho_j + \theta_j z_j + \lambda\} & \text{for } \theta_j > 0\end{cases}.$$

(v) From the second condition, we get that

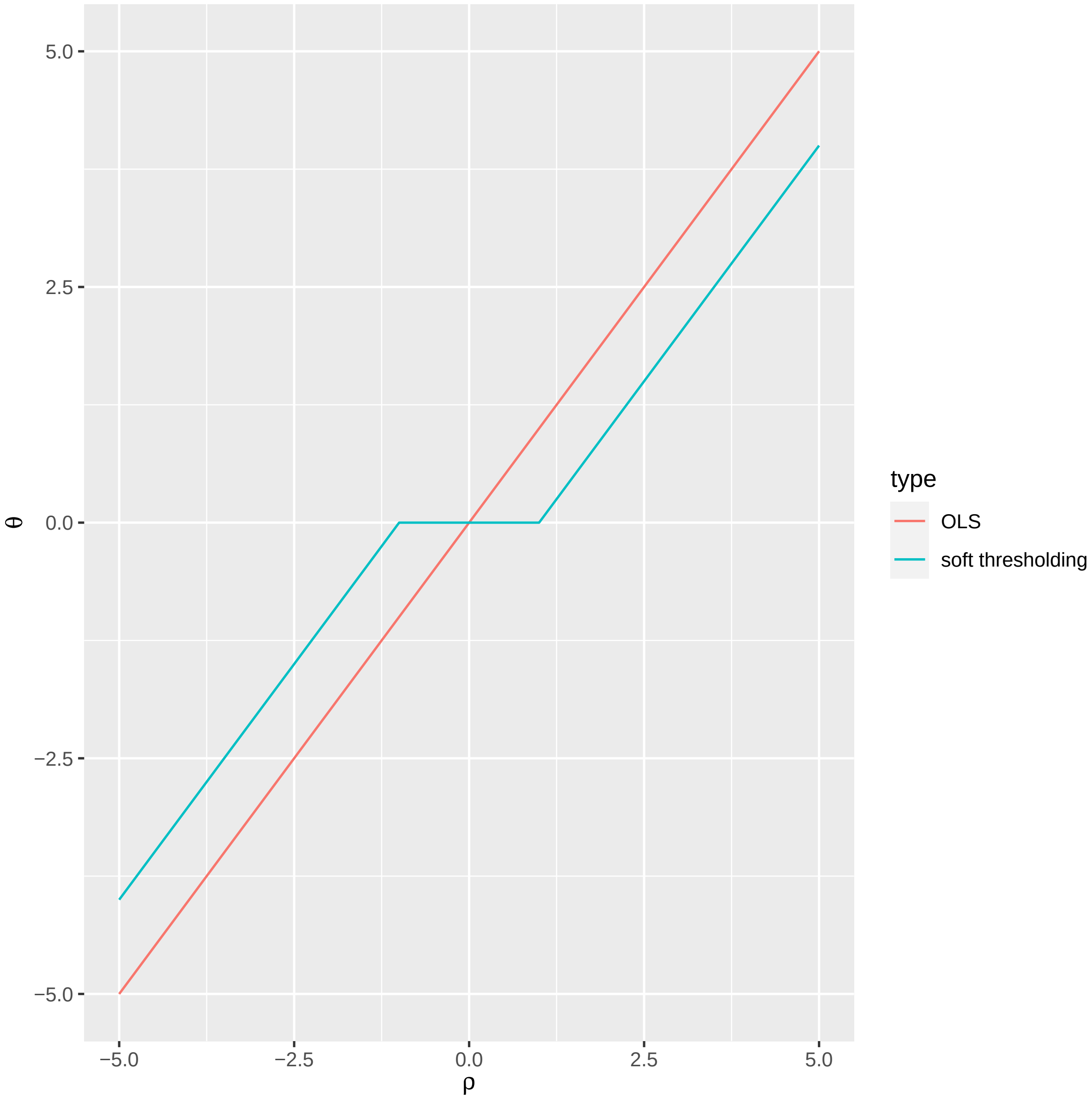
$$\begin{aligned}0 \in [-\rho_j - \lambda, -\rho_j + \lambda] &\iff -\rho_j - \lambda \leq 0 \leq -\rho_j + \lambda \\ &\iff -\lambda \leq \rho_j \leq \lambda.\end{aligned}$$

With this and setting the first and third conditions to zero and solving for θ_j we get that

$$\theta_j^* = \begin{cases}\frac{\rho_j + \lambda}{z_j} & \text{for } \rho_j < -\lambda \\ 0 & \text{for } -\lambda \leq \rho_j \leq \lambda \\ \frac{\rho_j - \lambda}{z_j} & \text{for } \rho_j > \lambda\end{cases}.$$

(vi)

```r
library(ggplot2)

rhos = seq(-5, 5, 0.1)
lambda = 1
z = 1
thetas_star = ifelse(rhos < -lambda, (rhos + lambda)/z,
                     ifelse(rhos > lambda, ((rhos - lambda)/z),
                            0))
thetas = rhos/z
df = rbind(data.frame(theta = thetas_star, type="soft thresholding", rhos=rhos),
      data.frame(theta = thetas, type="OLS", rhos=rhos))
ggplot(df) +
  geom_line(aes(x=rhos, y=theta, color=type)) +
  ylab(expression(theta)) +
  xlab(expression(rho))
```



(b) Since $\mathbf{X}^\top\mathbf{X}$ is non-singular, we know that $\mathbf{X}^\top\mathbf{X}$ must be positive definite by construction, i.e., there exist an orthogonal matrix $\boldsymbol{V}$ and a diagonal matrix $\boldsymbol{D}$ with $D_{ii} > 0$ such that

$$\boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^\top = \mathbf{X}^\top\mathbf{X}.$$

For $A = \boldsymbol{V}\boldsymbol{D}^{-0.5}$, we get that

$$(\boldsymbol{X}\boldsymbol{A})^\top\boldsymbol{X}\boldsymbol{A} = \boldsymbol{A}^\top\boldsymbol{X}^\top\boldsymbol{X}\boldsymbol{A} = \boldsymbol{D}^{-0.5}\boldsymbol{V}^\top\boldsymbol{V}\boldsymbol{D}\boldsymbol{V}^\top\boldsymbol{V}\boldsymbol{D}^{-0.5} = \boldsymbol{D}^{-0.5}\boldsymbol{D}\boldsymbol{D}^{-0.5} = \boldsymbol{I}.$$

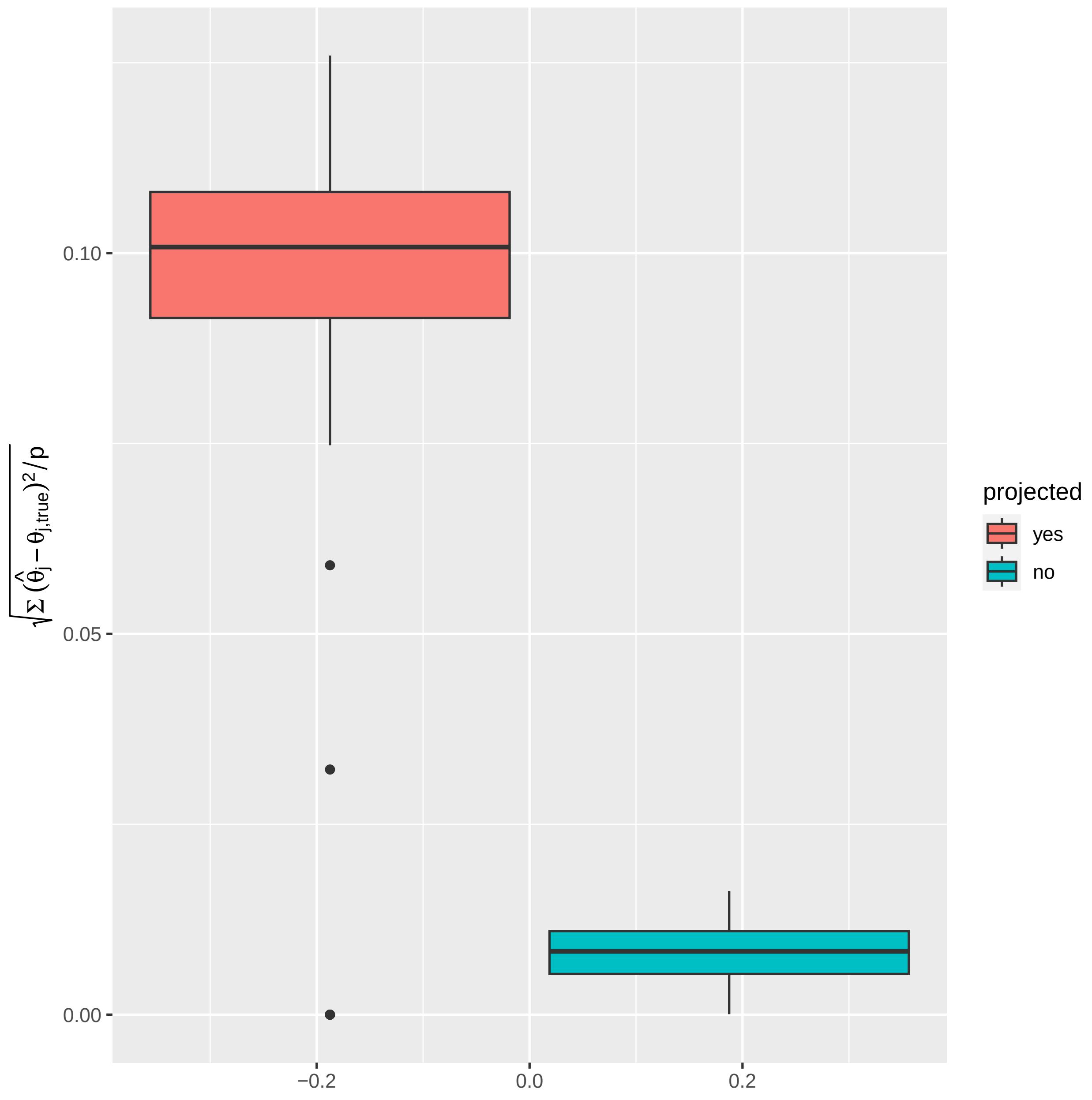
(c) (i) From $\nabla_{\boldsymbol{\theta}}\|\mathbf{X}\boldsymbol{A}\boldsymbol{\theta} - y\|_2^2 = 2\boldsymbol{A}\mathbf{X}^\top\mathbf{X}\boldsymbol{A}\boldsymbol{\theta} - 2\boldsymbol{A}^\top\boldsymbol{X}^\top y \overset{!}{=} 2\boldsymbol{X}^\top\mathbf{X}\boldsymbol{\theta} - 2\boldsymbol{X}^\top y = \nabla_{\boldsymbol{\theta}}\|\mathbf{X}\boldsymbol{\theta} - y\|_2^2$ it follows that in general $\boldsymbol{A} = \boldsymbol{I}$.

(ii) If we chose the penalty term to be $\|\boldsymbol{A}\boldsymbol{\theta}\|_1$ then this would result in $\arg\min_{\boldsymbol{\theta}\in\mathbb{R}^p} \mathcal{R}_{\text{reg}}(\boldsymbol{A}\boldsymbol{\theta})$ and since $\boldsymbol{A}$ is invertible by construction, this minimization would be equivalent to the original Lasso regression. However, because of the $\|\boldsymbol{A}\boldsymbol{\theta}\|_1$ term, it is generally not possible anymore to split the regularized risk into a sum of functions that only depend on one parameter.

(iii) We can see this procedure as a projection of the variables via $\boldsymbol{A}$ followed by a Lasso regression. Hence, we select variables in these projected coordinates, but this does not imply that the solution in the original coordinates $\boldsymbol{A}\boldsymbol{\theta}^*$ will be sparse.

(d)

```
library(matlib)
library(ggplot2)
set.seed(2)

proj_orth_lasso <- function(X, y, lambda){
  # compute X_tilde
  ev = eigen(t(X) %*% X)
  A = ev$vectors %*% diag(ev$values^(-0.5))
  X_tilde = X %*% A
  t(X_tilde) %*% X_tilde

  # compute analytical solution for X_tilde
  proj_theta_ols = t(X_tilde) %*% y
  proj_theta_star = sign(proj_theta_ols) * ifelse(abs(proj_theta_ols) - lambda > 0,
                                        abs(proj_theta_ols) - lambda, 0)
  theta_star = A %*% proj_theta_star
  return(c(theta_star))
}

lasso <- function(X, y, lambda, N){
  p = ncol(X)
  theta = rep(1.0, p)
  for(i in seq(N)){
    j = (i %% p)+1
    mask = rep(1, p)
    mask[j] = 0.0

    rho_j = X[,j] %*% (y - X %*% (theta * mask))
    z_j = sum(X[,j]^2)

    theta[j] = ifelse(rho_j < -lambda, (rho_j + lambda)/z_j,
                      ifelse(rho_j > lambda, (rho_j - lambda)/z_j, 0))
  }
  return(theta)
}
rmses = data.frame(rmse = numeric(), type = factor())

p = 10
n = 100

num_opt_steps=400

sigma_noise = 0.1
sigma_signal = 1.0

lambda = 1

rmses = data.frame(rmse = numeric(), projected = factor())

for(i in seq(100)){
  X = matrix(rnorm(n*p, sd=sigma_signal), nrow=n)
  theta_true = rnorm(p)
  idx = rbinom(p, 1, 0.7)
  theta_true[which(idx == 1)] = 0

  y = X %*% theta_true + rnorm(n, sd=sigma_noise)

  rmses = rbind(rmses, data.frame(rmse =
                    n/(n-1)*sd(proj_orth_lasso(X, y, lambda) - theta_true),
                    projected=factor("yes", levels=c("yes", "no"))))
  rmses = rbind(rmses, data.frame(rmse =
                    n/(n-1)*sd(lasso(X, y, lambda, num_opt_steps) - theta_true),
                    projected=factor("no", levels=c("yes", "no"))))
}

ggplot(rmses) +
  geom_boxplot(aes(y = rmse, fill = projected)) +
  ylab(expression(sqrt(Sigma~(hat(theta)[j]-theta["j,true"])^2/p)))
```



In this simulation, the true parameter vector is sparse in its original coordinate system. Hence, as expected, the regular Lasso regression outperforms the projected approach on average when identifying the true parameters in this scenario.